**Exercise 5: Triggers**

**Scenario 1: Automatically update the last modified date when a customer's record is updated.**

**o Question: Write a trigger UpdateCustomerLastModified that updates the LastModified column of the Customers table to the current date whenever a customer's record is updated.**

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)

VALUES (4, 'Test User', TO\_DATE('1990-01-01', 'YYYY-MM-DD'), 5000, SYSDATE-1);

COMMIT;

SELECT \* from Customers;

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

:NEW.LastModified := SYSDATE;

END;

/

UPDATE Customers SET Balance = Balance + 1000 WHERE CustomerID = 4;

COMMIT;

A table with numbers and numbers

AI-generated content may be incorrect.

**Scenario 2: Maintain an audit log for all transactions.**

**o Question: Write a trigger LogTransaction that inserts a record into an AuditLog table whenever a transaction is inserted into the Transactions table.**

CREATE TABLE AuditLog (

AuditID NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY PRIMARY KEY,

TransactionID NUMBER,

AccountID NUMBER,

TransactionDate DATE,

Amount NUMBER,

TransactionType VARCHAR2(10),

LogTimestamp DATE DEFAULT SYSDATE

);

select \* from Transactions;

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

INSERT INTO AuditLog (

TransactionID, AccountID, TransactionDate, Amount, TransactionType, LogTimestamp

)

VALUES (

:NEW.TransactionID,

:NEW.AccountID,

:NEW.TransactionDate,

:NEW.Amount,

:NEW.TransactionType,

SYSDATE

);

END;

/

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (3, 1, SYSDATE, 500, 'Deposit');

select \* from Transactions;

select \* from AuditLog;

A screenshot of a computer screen

AI-generated content may be incorrect.

**Scenario 3: Enforce business rules on deposits and withdrawals.**

**o Question: Write a trigger CheckTransactionRules that ensures withdrawals do not exceed the balance and deposits are positive before inserting a record into the Transactions table.**

CREATE OR REPLACE TRIGGER CheckTransactionRules

BEFORE INSERT ON Transactions

FOR EACH ROW

DECLARE

bal NUMBER;

BEGIN

BEGIN

SELECT Balance INTO bal

FROM Accounts

WHERE AccountID = :NEW.AccountID;

IF :NEW.TransactionType = 'Withdrawal' THEN

IF :NEW.Amount > bal THEN

DBMS\_OUTPUT.PUT\_LINE('Withdrawal exceeds balance, cannot proceed.');

END IF;

ELSIF :NEW.TransactionType = 'Deposit' THEN

IF :NEW.Amount <= 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Deposit amount must be positive.');

END IF;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Account ID does not exist.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Other error: ' || SQLERRM);

END;

END;

/

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (4, 1, SYSDATE, 500, 'Deposit');

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (5, 1, SYSDATE, 100000, 'Withdrawal');

A close-up of a message

AI-generated content may be incorrect.